AN ABSTRACT REDUCTION MODEL FOR COMPUTER SECURITY RISK

Mohamed Hamdi, Noureddine Boudriga
Computer Network and Security Research Lab.
SUP'COM, University of 7th of November, Carthage, Tunisia
mmh@certification.tn, nab@supcom.rnu.tn

Abstract This paper presents an approach for decision making under security risks in a computer network environment. The proposed method relies on a many sorted algebraic signature and on a rewriting system. This latter is shown to be terminating and yielding a normal form, called the risk analysis equation, that models the cost-benefit balance. Furthermore, a gradual algebraic resolution of the risk analysis equation is described.

This formalism helps security analysts to automate the selection of the optimal security solutions that minimize the residual risk.

Keywords: Risk management, algebraic specifications, rewriting systems, risk analysis equation.

1. INTRODUCTION

AS the use of information systems and sophisticated communication infrastructures becomes widespread in modern enterprises, the growth of various attacks against those systems is inevitable. In spite of the important investments made by companies to secure their assets, they are continuing to be the target of harmful actions. To this end, security ought to be considered as a part of the management process. A variety of Risk Analysis (RA) methods have been proposed in this context [Alberts and Dorofee, 2002; Stonebumer et al., 2002; GoC, 1996; GAO, 1999] . They aim at evaluating accurately the loss resulting from potential attacks in order to make the appropriate decisions. Nonetheless, those approaches still have numerous shortcomings. In fact, most of them rely on qualitative reasoning which is, by nature, non precise despite of its ease. For instance, defining a four or five-step scale to assess the probability of success of an attack does not allow a suitable differentiation between threats. Moreover, existing RA approaches have not involved a considerable theoretical development. They introduce several simple concepts (e.g., RA matrix) that are intended to make their manual application simpler. However, in a sys-

Y. Deswarte et al. (eds.), *Security and Protection in Information Processing Systems*

tem where hundreds (or even thousands) of assets, vulnerabilities, threats and decisions are considered, security management can not be addressed manually. Therefore, automated RA is a key concern which should be emphasized.

In this paper, we present a reduction system that permits to automate the reasoning made by security experts when performing RA. Our optimization framework consists of an algebraic rewriting system which yields the candidate security solutions and an algorithm for selecting the optimal countermeasure configuration through the use of an order relation on a lattice structure. The major merit of this work is that it allows to handle complicated situations where human intervention is not applicable. Furthermore, the algebraic framework includes a logic which offers the possibility to perform inferences and proofs. In addition, a formal proof of the efficiency of our approach is given through a demonstration of the termination of the rewriting system (using polynomial interpretations) and of the convergence of the algorithm. This shows that the system reaches the optimal security solutions independently from the situation.

The remaining part of the paper is organized as follows. Section 2 describes the RA signature representing the extension with a previous work [Hamdi and Boudriga, 2003]. Section 3 defines the rewriting system and studies its properties. Section 4 discusses the application of the concept of many-sorted algebras to RA. Section 5 proposes an algorithm to solve the RA equation. Section 6 concludes the paper.

2. THE RISK ANALYSIS SIGNATURE

2.1 Related work

Many-sorted signatures and first-order predicate logic have been previously used to model the RA problem [Hamdi and Boudriga, 2003]. Basically, a many-sorted signature Σ is characterized by a set S of sort names, an $S^{\star} \times S$-sorted set of operation names (denoted Ω), and an $S^{\star}$-sorted set of predicate symbols (denoted Π).

In [Hamdi and Boudriga, 2003], the authors have proposed the signature Σ_0 depicted in Table 1 to model the RA process. The richness of this algebraic signature allows the representation of attack scenarios (i.e., attacks performed on multiple steps). The introduction of this concept affects considerably the decision making process as the efficiency of a decision differs for two scenarios having two distinct semantic structures. This stems from the fact that the quantitative comparison attributes (e.g., probability of success, impact) of the main attack is computed using the attributes of the elementary attacks.

It can be demonstrated that security countermeasures can be viewed as pseudo-inverses of potential threats with respect to the composition law $\star$. More precisely, decisions aim at making the system recover from the effect of the various possible attacks. The major interest of this reasoning resides in

sig	$\Sigma_0=$	
	sorts	$\mathit{asset}, \mathit{vuln}, \mathit{attack}, \mathit{decision}$
	opns	$-\star- : \mathit{attack} \times \mathit{attack} \longrightarrow \mathit{attack}$
		$1_a :\longrightarrow \mathit{attack}$
		$(-,-)\bullet(-,-) : \mathit{decision} \times \mathit{asset} \times \mathit{decision} \times \mathit{asset} \longrightarrow \mathit{decision} \times \mathit{asset}$
		$a^\star :\longrightarrow \mathit{asset}$
	preds	$\mathit{ispresent} : \mathit{asset} \times \mathit{vuln}$
		$\mathit{exploits} : \mathit{attack} \times \mathit{vuln}$
		$\mathit{ispossible} : \mathit{attack} \times \mathit{asset}$
		$\mathit{isminimal} : \mathit{attack}$
		$- \leq_a - : \mathit{attack} \times \mathit{attack}$
		$\mathit{mitigates} : \mathit{decision} \times \mathit{attack}$
		$(-,-) \succ_c (-,-) : \mathit{decision} \times \mathit{asset} \times \mathit{decision} \times \mathit{asset}$

Table 1. Many-sorted signature representing the RA process.

the fact that the decision making process, which is the essence of RA, is performed through the resolution of the equation $a \star d = 1_a$, where a models the possible attacks, d represents the potential countermeasures and 1_a is the neutral element of $\star$. Despite its interesting properties, this algebraic framework does not allow a rigorous resolution of the above equation since the operation $\star$ can be applied only for two attacks to build a scenario. For instance, $at_1 \star at_2$ expresses a scenario where at_1 occurs before at_2. Therefore, the expression $a \star d$ is mathematically incorrect since the second term (i.e., the decision d in this case) should be of sort *attack*. Hence, the signature mentioned above should be extended in order to allow a more accurate representation of the RA problem that relies on the same basic idea.

2.2 A more general framework

Basically, RA consists in studying the environment of the target system in order to predict the potential threats and to derive the corresponding security solutions. To support this reasoning, the signature presented in [Hamdi and Boudriga, 2003] should be enriched in order to allow a convenient modeling of both the environment and the potential events. As these events can be either destructive or preventive, a balance between the potential threats and a set of security countermeasures should be considered. To this purpose, we introduce the following definition of a RA signature.

DEFINITION 1 ***Risk analysis signature.*** *A RA signature is a many-sorted algebra* $\Sigma = \langle S, \Omega, \Pi\rangle$ *that verifies the following conditions:*

(1) $\mathit{sysenv}, \mathit{action} \in S$,

(2) $\oplus : \mathit{sysenv} \times \mathit{sysenv} \longrightarrow \mathit{action} \in \Omega$,

(3) $\circledast : \mathit{sysenv} \times \mathit{action} \longrightarrow \mathit{action} \in \Omega$,

(4) $\otimes$: $action \times action \rightarrow action \in \Omega$,
(5) $\leq_{act}$: $action \times action \in \Pi$.

The alert reader would have noticed that this definition is more general than Σ_0 in the sense that it just specifies kernel representing of basic risk management concepts. Two special sorts, *sysenv* and *action*, have been considered to provide a formal view of the state of the analyzed system. Effectively, this state is characterized by the environment of the system (modeled by *sysenv*) and the actions that take it from one state to another (modeled by *action*). Furthermore, a special category of Σ-operations has been highlighted in this definition. They show the different methods that can be used to build actions. Three combinations to construct actions from the other sorts are provided. In fact, an action can be deduced purely from the system environment (using $\oplus$), or by taking into account a specific action (using $\circledast$). In addition, the combination of two actions can constitute an action (using $\otimes$). Finally, a predicate symbol $\leq_{act}$ has to be included in the signature so that actions can be compared.

A major advantage of this formal representation is that it translates perfectly the reasoning steps followed by the risk analyst which are: (1) the identification of the key components of the analyzed system, (2) the identification of the potential actions on the target system (i.e., attacks and countermeasures), (3) the selection of the best security decisions according to several preference criteria.

Table 2 gives an example of a RA signature. Four sorts are added to those mentioned in the above definition. Operations *ispresent*, *exploits*, *mitigates* and $\sqsubseteq$ give the environment information that the risk analyst should know at the first step. In fact, *ispresent* expresses the existence of a vulnerability inside a given resource while *exploits* tells whether a given vulnerability should be present to carry out a corresponding attack. On the other hand, *mitigates* means that a specified decision reduces, in a certain manner, the effect of a given attack. The last operation $\sqsubseteq$ states that an attack is a part of a more global attack scenario (i.e., the first attack is a sub-scenario of the second).

The second class of operators are related to the construction of attack scenarios. The composition law $\star$ is a precedence operator used to build composite attacks. The constant 1_a stands for the null attack which correspond to "*no action*" on the system.

3. THE REWRITING SYSTEM

Having built a RA signature, its properties must be correctly expressed in order to conduct an appropriate reasoning from RA point of view. Typically, this involves a set of axioms (or equations) and a corresponding inference system which derives all possible consequences from those axioms. In [Hamdi and

sig	$\sum$=	
	sorts	$asset$, $vuln$, $attack$, $decision$, $sysenv$, $action$
	opns	$ispresent: asset \times vuln \longrightarrow sysenv$
		$exploits: attack \times vuln \longrightarrow sysenv$
		$mitigates: decision \times attack \longrightarrow sysenv$
		$\sqsubseteq$: $attack \times attack \longrightarrow sysenv$
		$- \star -: attack \times attack \longrightarrow attack$
		$1_a :\longrightarrow attack$
		$\oplus : sysenv \times sysenv \longrightarrow action$
		$\circledast : sysenv \times action \longrightarrow action$
		$\otimes : action \times action \longrightarrow action$
		$1_{act} :\longrightarrow action$
		$\diamond : attack \times asset \longrightarrow action$
		$\circ : decision \times asset \longrightarrow action$
	preds	$\leq_{act}$: $action \times action$

Table 2. Example of a RA signature.

Boudriga, 2003], we used a first-order predicate logic and a classical Gentzen system to automate the deduction process. However, as this approach relies on replacement of equals by equals, it requires a special attention. In fact, knowing that $at \star 1_a = 1_a$ (i.e., 1_a is the neutral element on the set of attacks), the term at_1 in the equation $at_3 = at_1 \star at_2$ can be replaced by $at_1 \star 1_a$ leading to the equation $at_3 = at_1 \star 1_a \star at_2$. This does not conform with the risk analyst's objective which is to go through several deduction steps towards an equation representing the balance between attacks and security decisions. It turns out that rewriting systems (composed of directed equations) provide this possibility. In this section, we present a specific class of rewriting systems that is suitable for RA problems and we discuss its main properties.

3.1 Defining the rewriting rules

Term rewriting systems are widely used in the field of formal modeling. Their main feature is that they give the ability to automate the generation of canonical terms. Typically, a rewriting system $\langle T(\Sigma, \chi), \longrightarrow\rangle$ is a finite set of rewriting rules which are ordered pairs of terms denoted $\tau \longrightarrow \tau'$, where $\longrightarrow$ is a binary relation on $T(\Sigma, \chi)$ (the set of Σ-terms). A term t rewrites to a term t', denoted by $t \longrightarrow t'$, if there exists a rule $\tau \longrightarrow \tau'$ in $\langle T(\Sigma, \chi), \longrightarrow\rangle$, a position ω in t, a substitution σ, satisfying $t_{|\omega} = \sigma(\tau)$, such that $t' = t\left[\sigma(\tau')\right]_{\omega}$. A term t is called a normal form if there is not t' such that $t \longrightarrow t'$ (t is then denoted by $t \downarrow_{\langle T(\sum,\chi),\longrightarrow\rangle}$ in this case).

(ρ_1)	$ispresent(v,as) \oplus exploits(at,v) \longrightarrow at \diamond as$
(ρ_2)	$mitigates(d,at) \circledast at \diamond as \longrightarrow at \diamond as \otimes d \circ as$
(ρ_3)	$(at_1 \diamond as) \otimes (at_2 \diamond as) \longrightarrow (at_1 \star at_2) \diamond as$
(ρ_4)	$at_1 \sqsubseteq at2 \circledast (at_1 \diamond as \otimes at_2 \diamond as) \longrightarrow at_1 \diamond as$
(ρ_5)	$at \star 1_a \longrightarrow at$
(ρ_6)	$1_a \star at \longrightarrow at$
(ρ_7)	$act \otimes act \longrightarrow act$

Table 3. Example of a RA rewriting system.

In our context, we build a rewriting system $\rho = \langle T(\Sigma, \chi), \longrightarrow_\rho \rangle$ which simulates the reasoning of the human risk analyst. It allows the representation of the state of the analyzed system suitably for the decision selection process.

DEFINITION 2 ***Risk analysis rewriting system.*** *Let $\Sigma = \langle S, \Omega, \Pi \rangle$ be a RA signature. A RA rewriting system $\rho = \langle T(\Sigma, \chi), \longrightarrow_\rho \rangle$ is a set of rewriting rules which have one of the following forms:*

(i) Type I rules: $\tau \longrightarrow \tau'$ such that $\tau, \tau' \in T(\Sigma, \chi)$ and τ' is a sub-term of τ,

(ii) Type II rules: $\tau \longrightarrow \tau'$ such that $\tau, \tau' \in T(\Sigma, \chi)$, τ contains one function symbol and τ' does not contain any function symbol,

(iii) Type III rules: $\tau\varphi\tau' \longrightarrow \tau''\varphi'\tau'''$ such that $\tau, \tau', \tau'', \tau''' \in T(\Sigma, \chi)$ and $\varphi, \varphi' \in \{\oplus, \otimes, \circledast\}$.

This definition asserts that rewriting rules expressing RA reasoning are restricted to three types. This may seem constraining to the security specialist but we will show through some examples that RA rewriting systems are sufficiently general to allow an efficient representation of a practical context. For instance, the system ρ_0 given in Table 3 is a RA rewriting system. In fact, the reader can easily check that (ρ_4) is a type I rule ($at_1 \diamond as$ is a sub-term of $(at_1 \diamond as \otimes at_2 \diamond as) \otimes at_1 \sqsubseteq at2$) and that $\{(\rho_5), (\rho_6), (\rho_7)\}$ are type II rules. In addition, $\{(\rho_1), (\rho_2), (\rho_3)\}$ are type III rules.

The first rule shows how to state whether a threat is actually possible to perform on the system. In fact, this threat has to exploit a vulnerability which effectively exists in the corresponding asset.

The rule (ρ_2) is a heuristic that reduces the space where the security analyst searches for the optimal countermeasures. Instead of considering the whole set of security solutions, only those which potentially mitigate the effect of the possible attacks should be addressed. The two following rewriting rules are related to attacks scenarios. (ρ_3) states that if two attacks are possible to carry out on an asset, then so does their composition. On the other hand, (ρ_4) gives that if several attacks are considered as potential actions, then the system would focus on the one that includes the others (in the sense of the operator $\sqsubseteq$). Rules (ρ_5) and (ρ_6) simply mean that 1_a is the neutral element of the operation $\star$. The last rule permits to avoid redundancies when addressing potential attacks or candidate security solutions.

3.2 Termination and confluence of the rewriting system

The two main properties of a rewriting system are confluence and termination. A rewriting rule $\rightarrow$ is said to be confluent if it satisfies $^{\star} \leftarrow \circ \rightarrow^{\star} \subseteq \rightarrow^{\star} \circ^{\star} \leftarrow$, where $\leftarrow$ denotes the inverse of $\rightarrow$, and $\circ$ and * represent respectively the composition and the transitive closure of binary relations. On the other hand, termination means that there is not infinite chains of related elements $t_1 \rightarrow t_2 \rightarrow ... \rightarrow t_i \rightarrow ...$

Confluence means that the order of application of the rewriting rules to a given term is not important. In other terms, two normal forms obtained through applying two distinct sequences of rewriting rules to the same term must be equal. This property is not important in our case since the order that should be followed is known *a priori*. In fact, a typical scenario-based RA consists of the following steps: (1) threat identification, (2) attack scenario identification, (3) candidate security solutions identification, and (4) countermeasure selection. This situation is relatively easy to handle since operations are clearly ordered within each step. In other contexts where concurrent rewriting rules may be applied to achieve the same task, confluence would be seriously considered.

On the opposite, termination is an essential property in our context as it guarantees that the system reaches the normal forms, which is an major requirement from the RA point of view. In this paper, we adopt polynomial interpretations as a tool to demonstrate termination. This consists in assigning to each Σ-operation a polynomial verifying three basic requirements: (1) for each Σ-operation o of arity (i.e., number of variables) n, a polynomial of the same arity, denoted $[o]\,(X_1, ..., X_n)$, is associated, (2) the variables $X_1, ..., X_n$ of any polynomial of arity n are supposed to be integers greater than or equal to a constant c, and (3) all polynomials should have non-negative integers coefficients.

This polynomial representation can be extended to terms by induction. For example, if $[\star]\,(X_1, X_2) = X_1 + X_2$ and $[\sqsubseteq]\,(X_1, X_2) = X_1.X_2$, then the interpretation of $at_1 \sqsubseteq (at_1 \star at_2)$ is expressed by $X_1.\,[\star]\,(X_1, X_2)$, which is equal to $X_1.(X_1 + X_2)$. Thus, a polynomial function $[t]\,(X_1, ..., X_n)$ can be mapped to every $t \in T\left(\sum, \chi\right)$. Moreover, an order relation on $T\left(\Sigma, \chi\right)$, denoted by $<_{[.]}$, can be set through the use of these interpretations such that $t <_{[.]} t'$ if, and only if $[t] < [t']$ for every two terms $t, t' \in T\left(\Sigma, \chi\right)$, where $<$ is the order relation on positive integers.

In [B. Cherifa and Lescanne, 1987], it has been shown that this order is stable by instantiation and that it can therefore be used to prove the termination of rewriting systems. In fact, to terminate, every rule $\tau \longrightarrow \tau'$ of the rewriting system has to verify $[\tau] < [\tau']$. This method can be applied to RA rewriting systems.

THEOREM 3 *Let* $\Sigma = \langle S, \Omega, \Pi \rangle$ *be a RA signature and* ρ *be a RA rewriting system on* Σ *consisting only of type III rules.*

Let $\rhd$ *be a binary relation on* $T(\Sigma, \chi)$ *defined as follows:*

$\tau_1 \rhd \tau_2$ *iff.* $\exists \varphi, \varphi' \in \{\oplus, \otimes, \circledast\}$, $\tau_3, \tau_4 \in T(\Sigma, \chi)$ *st.* $(\tau_1 \varphi \tau_3 \longrightarrow \tau_2 \varphi' \tau_4) \in \rho$.

If $\rhd$ *is non-reflexive, non-symmetric and transitive then the rewriting system* ρ *terminates.*

Proof. To the demonstration purpose, we associate the polynomial interpretation $[\tau_0] = \sum_{\{\tau | \tau_0 \rhd \tau\}} [\tau]$ to each term τ_0 appearing in type III rules. Since $\rhd$ is non-reflexive, non-symmetric and transitive, it can be concluded that $\{\tau \mid \tau_2 \rhd \tau\} \subset \{\tau \mid \tau_1 \rhd \tau\}$ if $\tau_1 \rhd \tau_2$. Which implies:

$$\text{if } \tau_1 \rhd \tau_2 \text{ then } [\tau_1] > [\tau_2]. \tag{1}$$

We assume now that the polynomial interpretation $[\varphi](X_1, X_2) = X_1 + X_2$ correspond to every function φ in $\{\otimes, \oplus, \circledast\}$. Thus, for every type III rewriting rule $\tau_1 \varphi \tau_2 \longrightarrow \tau_3 \varphi' \tau_4$, we have to prove that $[\tau_1] + [\tau_2] > [\tau_3] + [\tau_4]$. This can be achieved for a rewriting rule $\tau_1 \varphi \tau_2 \longrightarrow \tau_3 \varphi' \tau_4$, using Equation 1 and the fact that $\tau_1 \rhd \tau_3$,$\tau_2 \rhd \tau_4$.□

From the above theorem, it can be concluded that a system composed of type III rewriting rules terminates if it verifies:

If τ_1 *appears in the left term of a rewriting rule and* τ_2 *appears in the right term of the same rule, and if* τ_1*appears in the right term of an other rewriting rule, then* τ_2 *does not appear in the left term of the latter rule.*

EXAMPLE 4 ***Counter-example of termination****. The above theorem gives a sufficient condition for termination of type III rewriting rules. Obviously, if a specific system does not fulfill this condition, we can not conclude that it does not terminate. Nonetheless, we discuss the following example to show how this condition can affect the termination.*

Consider the RA rewriting system consisting of the two following type III rewriting rules:

$$\begin{cases} (\rho_1) \ mitigates(d, at) \circledast at \diamond as \longrightarrow at \diamond as \otimes d \circ as, \\ (\rho_2) \ at \diamond as \circledast d \circ as \longrightarrow mitigates(d, at). \end{cases}$$

It is clear that this system does not verify the assumption of the above theorem as (ρ_1) *gives that* $mitigates(d, at) \rhd d \circ as$ *and it comes from* (ρ_2) *that* $d \circ as \rhd mitigates(d, at)$. *In addition, the system does not terminate because the rewriting sequence* $\rho_1, \rho_2, \rho_1, \rho_2 ...$ *is infinite cycle.*

$[\mathit{ispresent}]\,(X_1, X_2) = X_1 + X_2,$	$[\mathit{exploits}]\,(X_1, X_2) = X_1 + X_2,$
$[\mathit{mitigates}]\,(X_1, X_2) = X_1 + X_2,$	$[\sqsubseteq]\,(X_1, X_2) = X_1.X_2,$
$[\star]\,(X_1, X_2) = X_1 + X_2,$	$[1_a] = 2,$
$[\oplus]\,(X_1, X_2) = X_1.X_2,$	$[\circledast]\,(X_1, X_2) = X_1.X_2 + X_2 + 2,$
$[\otimes]\,(X_1, X_2) = X_1 + X_2 + 2,$	$[1_{act}] = 2,$
$[\diamond]\,(X_1, X_2) = X_1.X_2,$	$[\circ]\,(X_1, X_2) = X_1.X_2.$

Figure 1. Polynomial interpretations corresponding to Σ-operations.

$P_{\rho_1}(X_1, X_2, X_3) = X_1^2 + X_1.X_2 + X_1.X_3$
$P_{\rho_2}(X_1, X_2, X_3) = X_1.X_3.(X_2 - 1) + X_2^2.X_3$
$P_{\rho_3}(X_1, X_2, X_3) = 2, P_{\rho_4}(X_1) = X_1.X_3 + X_2.X_3 + 4$
$P_{\rho_5}(X) = 2, P_{\rho_6}(X) = 2, P_{\rho_7}(X) = X + 2$

Figure 2. Polynomials corresponding to the rewriting system ρ_0.

THEOREM 5 *Let $\Sigma = \langle S, \Omega, \Pi \rangle$ be a RA signature and ρ be a RA rewriting system on Σ. If the sub-system composed of the type III rules of ρ verifies the condition of theorem 3, then ρ terminates.*

Particularly, the rewriting system ρ_0 terminates.

This theorem is a direct consequence of Theorem 3 since type I and type II rewriting rules terminate trivially (see [Goguen and Malcolm, 2000; Loeckx et al.] for more details). Concerning the system ρ_0, we use a set of polynomial interpretations which is different from the one mentioned in the above theorem. This allows a better illustration of the use of polynomial interpretations as a tool for proving termination of rewriting systems. Figure 1 associates an interpretation to each function of Σ.

The polynomials corresponding to the rules of our rewriting system are given in Figure 2. They have been computed using the induction principle discussed above. It is clear that all of these polynomials are positive if the variables are greater than 2 (i.e., $c = 2$). For the sake of simplicity, the computational steps to reach $(P_{\rho_i})_{i \in \{1,..,7\}}$ are not given. We leave it to the reader to check them.

4. TOWARDS RISK ANALYSIS ALGEBRAS

4.1 From specification to algebra

A many-sorted algebra assigns a concrete aspect to a many-sorted signature by associating a set of data to each sort and a function to each operation. In other terms, if $\Sigma = \langle S, \Omega, \Pi \rangle$ is a many-sorted signature, a Σ-algebra α assigns: (1) a set $|\alpha|_s$ to each sort $s \in S$, called the carrier set of the sort s (as defined in [Loeckx et al.]), (2) a function $|f|_\alpha : |\alpha|_{s_1} \times ... \times |\alpha|_{s_k} \rightarrow |\alpha|_s$ to each operation $(f : s_1 \times ... \times s_k \rightarrow s) \in \Omega$, (3) a predicate $|p|_\alpha : |\alpha|_{s_1} \times ... \times |\alpha|_{s_k}$ to each predicate symbol $(p : s_1 \times ... \times s_k) \in \Pi$. In our case, we can state that if Σ is a RA signature then α simply represent a RA project consisting of a real analyzed system and of a knowledge database characterizing the risk analyst.

In this section, we show how a RA rewriting system can be used to perform deductions that translate the representation of the studied information system from a complex equation to a normal form. To this purpose, we propose a general reduction methodology composed of the following steps:

1 Model the state of the system through the use of functions having the form $|f|_\alpha : |\alpha|_{s_1} \times ... \times |\alpha|_{s_k} \rightarrow |sysenv|_s$.

2 Apply rewriting rules that combine terms of sort $sysenv$ and generate the related terms of sort $action.$

3 Apply rewriting rules that combine terms of sort $sysenv$ and terms of sort $action$ to generate terms of sort $action.$

4 Apply rewriting rules that combine terms of sort $action$ and generate terms of the same sort. These latter terms should be normal forms with respect to the used rewriting system.

This methodology allows the automated risk analyst to perform a progressive reasoning leading to a term representing the actions of interest (i.e., destructive and preventive actions). A key feature of this term is that it consists of two different parts. A static part corresponds to destructive actions. It is called *static* because the risk analyst can not act on them, he can only introduce other actions that reduce their effect. On the other hand, a variable part represents preventive actions. The RA decision maker should select some of these actions (i.e., a sub-part of the variable part) to provide the best security level for the analyzed networked system.

For instance, in the case of the rewriting system presented in the previous section, the normal form generated by our methodlogy is expressed by the following equation:

$$\left(\left(|\otimes|_\alpha\right)_{i \leq n,\, j \leq n_i} (at_j \diamond as_i)\right) \left(|\otimes|_\alpha\right) \left(\left(|\otimes|_\alpha\right)_{i \leq n,\, j \leq m_i} (d_j \circ as_i)\right). \qquad (2)$$

This term will be used in Section 5 in order to search for the optimal countermeasures combination. For the seek of clarity, the subscript α will be removed from sorts and operations when no confusion can be made about the many-sorted RA algebra.

4.2 Illustrative example

The goal of this subsection is to illuminate the key concepts presented above at a sufficient level to ensure a fundamental understanding of their usage in practice. The first step is to build a many-sorted algebra α that conforms with the specification $\langle \Sigma_0, \rho_0 \rangle$. This means that a set $|\alpha|_s$ is associated to every sort s in the signature Σ_0.

We suppose that the analyzed system is composed of two assets, say as_1 and as_2 (i.e. $|\alpha|_{asset} = \{as_1, as_2\}$), that verify the hypotheses below:

(H_1) as_1 and as_2 are placed behind a packet filtering gateway that prevent connection establishment from the external network.

(H_2) The company has no security awareness program .

(H_3) The user working on as_1 introduced a strong root password.

(H_4) The user working on as_2 did put his name as root password for this machine.

(H_5) The FTP (File Transfer Protocol) port (21) is closed in both as_1 and as_2.

Furthermore, it is assumed that the basic knowledge of the RA system consists of the following attacks and vulnerabilities ((a_i) denote attacks and (v_i) stand for vulnerabilities):

(a_1) Execute remotely on the victim machine Netcat, a Unix utility which reads and writes data across network connections.

(a_2) Perform a SYN (synchronization) scan. This consists in sending a SYN packet to every port on the victim machine and waiting for responses. If the victim acknowledges the packet, then the port is open.

(a_3) Perform an ACK (acknowledgment) scan. This technique is more sophisticated as it permits to know if a port is open even it is protected through a packet filter.

(a_4) Get password files from the victim machine. For instance, in a Unix environment, the attacker gets the /etc/passwd and /etc/shadow.

(a_5) Perform a dictionary attack on the root password. This attack can be performed by testing all the combinations generated from a wordlist.

(a_6) Perform a buffer overflow on FTP server application to get remote root access to the victim machine.

(v_1) The user executes programs coming from non-trusted sources.

(v_2) The host responds to SYN scan. This vulnerability can be removed using a packet filtering firewall that blocks incoming connections.

(v_3) The host responds to ACK scan. Packet filtering firewalls are not efficient to overcome this weakness. Stateful inspection firewall, which monitor the state of TCP connections, can be used to cover this limit.

(v_4) Weak root password. This means that the password can be easily cracked (e.g., brute force, dictionary).

(v_5) FTP port is open. Many FTP server implementations contain security breaches, especially buffer overflow vulnerabilities.

In addition, two attacks scenarios, denoted a_7 and a_8 are considered. They are expressed by the following equation:

$$a_7 = a_1 \star a_2 \star a_4 \star a_5, \;\; a_8 = a_1 \star a_3 \star a_4 \star a_5. \tag{3}$$

These scenarios correspond to two alternatives of the same main attack (password cracking), the first one relies on SYN scan while the second one is based on ACK scan. In other terms, $|\alpha|_{attack} = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$.

In addition to these sets, we use binary relations to represent operations and predicates of the signature $\sum$ in the algebra α. For instance, the following matrix models the relation which states whether a vulnerability is exploited by a given attack.

$$|exploits|_\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ - & - & - & - & - \\ - & - & - & - & - \end{bmatrix}.$$

If the value of $(|exploits|_\alpha)_{ij}$ equals 1 (respectively 0), this means that the attack a_i exploits (respectively does not exploit) the vulnerability v_j. The rows corresponding to a_7 and a_8 can not be filled because those attacks are composite scenarios.

In the following, we show how the oriented equations of ρ_0 can model efficiently two fundamental RA steps: vulnerability analysis and threat analysis.

Vulnerability identification

The conduction of the vulnerability identification process through the use of automated scanners and questionnaires, results in the following matrix (with regard to the hypotheses $(H_i)_{i \in \{1,..,5\}}$).

$$|ispresent|_\alpha = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Threat identification

As it has been mentioned above, this step aims at identifying the attacks that are possible to carry out on the analyzed system. In fact, a direct application of the rewriting rule (ρ_1) states that this operator can be modeled by a binary relation which is the composition of $|ispresent|_\alpha$ and $|exploits|_\alpha$. Thus, $\otimes$ appears in α as a binary relation denoting the composition between relations. More formally, we have:

$$|\diamond|_\alpha = |exploits \otimes ispresent|_\alpha = |exploits|_\alpha \circ |ispresent|_\alpha .$$

Consequently, the matrix corresponding $|\diamond|_\alpha$ is the product of $|ispresent|_\alpha$ and $|exploits|_\alpha$. Nonetheless, it is worth noting that the two bottom rows can not be computed through the use of this product as attack scenarios are not considered in the rewriting rule (ρ_1). Hence, the rule (ρ_4) has been applied to compute $(|\diamond|_\alpha)_{(i,j)\in\{7,8\}\times\{1,2\}}$.

$$(|\diamond|_\alpha)_{(i,j)\in\{7,8\}\times\{1,2\}} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Candidate countermeasure selection

We suppose that the following actions are available to reduce the effect of potential attacks.

(d_1) Acquire a stateful inspection firewall.

(d_2) Configure the packet filtering firewall to close FTP connections.

(d_3) Elaborate a security training program for the employees.

(d_4) Change root password to guarantee more robustness.

It is obvious that only (d_1), (d_2) and (d_3) mitigate the attack scenario as they thwart respectively (at_3), (at_1) and (at_5). Therefore, according to the rewriting rule (ρ_2), these decisions constitute the set of candidate countermeasures for the resource as_2. Concerning as_1, it is not taken into account during this step because it was found that no attack scenario is possible to carry out against it. The normal form can then be represented by the following formula:

$$at_8 \, |\diamond|_\alpha \, as_2 \otimes d_1 \, |\circ|_\alpha \, as_2 \otimes d_3 \, |\circ|_\alpha \, as_2 \otimes d_4 \, |\circ|_\alpha \, as_2. \tag{4}$$

5. SOLVING THE RISK ANALYSIS EQUATION

Having applied the rewriting rules according to the aforementioned order, we obtain an irreducible formula (see Equation 2). This formula expresses the potential destructive and preventive actions that correspond to the current state of the system. Since the aim of the security analyst is to thwart the attacks threatening the computer network, the countermeasure selection process can be thought of as the resolution of the equation:

$$\bigotimes_{i\leq n,\, j\leq n_i} (at_j \diamond as_i) \bigotimes \bigotimes_{i\leq n,\, j\leq m_i} (d_j \circ as_i) = 1_{act}, \tag{5}$$

meaning that the chosen security solutions totally annihilate the effect of the possible threats. However, since perfect security is an utopia, a realistic objective would be to select the decisions that make the effect of the above formula closest to the one of 1_{act}.

The resolution of this problem can be viewed as the selection of an optimal subset of the set $\Delta = \bigcup_{i\leq n, j\leq m_i} \{(d_j \circ as_i)\}$ of candidate decisions. Elements

$(d_j \circ as_i)$ of Δ will be denoted by δ. Obviously, the set $\Delta^\star$ (set of partitions of Δ) has a lattice structure with respect to the inclusion binary relation $\subset$. The universal lower bound of this lattice is $\emptyset$ (the empty set) while the universal greater bound is Δ itself. The optimization process can be thought of as a traverse of the lattice $(\Delta^\star, \subset)$ from $\emptyset$ towards the direction of Δ. We propose the following algorithm to select the optimal set of decisions:

```
Algorithm decision_select
Begin
$\delta^\star := 1_{act};$
$\Delta^{opt} = \emptyset;$
While $\left( \bigotimes_{i,j} (at_j \diamond as_i) \otimes \bigotimes_{\delta \in \Delta^{opt}} (\delta) \right) \leq_{act} \left( \bigotimes_{i,j} (at_j \diamond as_i) \otimes \bigotimes_{\delta \in \Delta^{opt} \cup \{\delta^\star\}} (\delta) \right)$
    $\Delta^{opt} := \Delta^{opt} \cup \{\delta^\star\};$
    Find $\delta^\star \in Parent(\Delta^{opt})$ such that for every $\delta_0 \in Parent(\Delta^{opt})$
    $\left( \bigotimes_{i,j} (at_j \diamond as_i) \otimes \bigotimes_{\delta \in \Delta^{opt} \cup \{\delta^\star\}} (\delta) \right) \leq_{act} \left( \bigotimes_{i,j} (at_j \diamond as_i) \otimes \bigotimes_{\delta \in \Delta^{opt} \cup \{\delta_0\}} (\delta) \right)$
EndWhile
End
```

where $Parent(X)$ represents the set $\{Y | Y \in \Delta^\star,\ X \subset Y \text{ and } |Y| = |X| + 1\}$ for every $X \in \Delta^\star$.

An informal reading of this algorithm shows that the optimal set of countermeasures ($\Delta^{opt} \subset \Delta^\star$) is reached through an iterative process which consists basically in adding, at each iteration, a security decision and evaluating its effect. At the beginning, the set Δ^{opt} is empty. Then, at each step, a solution $\delta^\star$ is selected from $Parent(\Delta^{opt})$ such that its combination with the elements of Δ^{opt} and the set of possible attacks be as close as possible to 1_{act} (with respect to $\leq_{act}$). The algorithm stops when the addition of any element $\delta \in \Delta \setminus \Delta^{opt}$ worsens the situation defined by Δ^{opt}.

THEOREM 6 *If the operation $\otimes$ is compatible with $\leq_{act}$ then the algorithm* **decision_select** *is $\leq_{act}$-optimal (i.e., every iteration gives a solution which is better than the solutions found at the previous iterations).*

Proof. We proceed to an *ab absurdo* proof. Supposing that the decisions δ_1 and δ_2 have been selected after the two first iterations of **decision_select** and that there exists $\delta_i, \delta_j \in \Delta \setminus \{\delta_1, \delta_2\}$ such that their combination is better than $\delta_1 \otimes \delta_2$:

$$\delta_i \otimes \delta_j \leq_{act} \delta_1 \otimes \delta_2. \tag{6}$$

As $\leq_{act}$ is compatible with $\otimes$ and $\delta_1 \leq_{act} \delta_i$, we can write:

$$\delta_1 \otimes \delta_j \leq_{act} \delta_i \otimes \delta_j. \tag{7}$$

Moreover, as δ_2 is the optimal decision at the second step:

$$\delta_1 \otimes \delta_2 \leq_{act} \delta_1 \otimes \delta_j \leq_{act} \delta_i \otimes \delta_j,$$

which conflicts with the main assumption expressed by Equation 6. □

Furthermore, the algorithm terminates because the search space at a given iteration is strictly included in the search space of the previous iteration. This means that the search space, which is finite, becomes more restricted across iterations.

A comprehensive example is given in the following to illustrate this algorithm. It is based on the concrete case studied in Section 4. The normal form reached at the end of that section (Equation 4) gives that the set of security decisions is equal to $\Delta = \{d_1 \circ as_2, d_3 \circ as_2, d_4 \circ as_2\} = \{\delta_1, \delta_2, \delta_3\}$. We propose to assess the cost and the benefit (in monetary terms) associated to each element $\delta \in \Delta$ and to state that $\delta \leq_{act} \delta'$ if, and only if, $(benefit - cost)_\delta \geq (benefit - cost)_{\delta'}$. We assume that the cost of δ_1, δ_2 and δ_3 are respectively equal to 1000, 400 and 30. On the other hand, the three candidate decisions have the same benefit, equal to 500, because they mitigate the same attack scenario.

Consequently, a direct application of the aforementioned algorithm gives that δ_3 is selected at the first step of **decision_select** because $(benefit - cost)_{\delta_1} = -500$, $(benefit - cost)_{\delta_2} = 100$ and $(benefit - cost)_{\delta_3} = 470$. At the second step, we evaluate the countermeasure combinations that belong to $Parent(\{\delta_3\})$ (i.e., $\{\delta_1, \delta_3\}$ and $\{\delta_2, \delta_3\}$). It is easy to check that $(benefit - cost)_{\delta_1 \otimes \delta_3} = -530$ and $(benefit - cost)_{\delta_2 \otimes \delta_3} = 70$. Therefore, no combination is selected at this level since both candidate decisions worsen the state reached at the previous iteration. Therefore, the final selected set of security decisions is $\{\delta_3\}$ according to the algorithm **decision_select**. In fact, the cost of δ_3 is acceptable while the stateful inspection firewall and the training are too expensive with respect to the attack that it prevents.

6. CONCLUSION

In this paper, we developed an algebraic decision making approach under security risks in a networked environment. Our method consists of two basic steps: a rewriting system that allows to identify the candidate countermeasures and a selection algorithm. We proved the termination of both of these steps meaning that the approach does not diverge.

Our work can be improved in the future by enriching the rewriting system (involving a study of its confluence) through the addition of a business-oriented reasoning.

References

Alberts, C.J. Dorofee, A.J. (2002).Managing Information Security Risks: the OCTAVE Approach, *Addison Wesley Professional*, ISBN: 0321118863.

Ben Cherifa, A. Lescanne, P. (1987). Termination of Rewriting Systems by Polynomial Interpretations and its Implementation, *Science of Computer Programming*, 9(2):137-160.

Claßen, I. Ehrig, H. Wolz, D. Algebraic Specification Techniques and Tools for Software Development: the ACT Approach, *AMAST Series in Computing (1)*, ISBN: 981-02-1227-5.

Goguen, J.A. Malcolm, G. (2000). Software Engineering with OBJ: Algebraix Specification in Action, *Kluwer Academic Publishers*, Boston, ISBN: 0-7923-7757-5.

Hamdi, M. Boudriga, N. (2003). Algebraic Specification of Network Security Risk Management, *First ACM Workshop on Formal Methods in Security Engineering*, Washington D.C.

Loeckx, J. Ehrich, H-D. Wolf, M. "Specification of Abstract Data Types," *Wiley and Teubner*, ISBN: 0-471-95067-X.

Stonebumer, G. Grogen, A. Fering, A. (2002). Risk Management Guide for Information Technology Systems, *National Institute for Standards and Technology*, Special Publication 800-30.

A Guide to Risk Management and Safeguard Selection for IT Systems, *Government of Canada, Communications Security Establishment*, January 1996.

Information Security Risk Assessment: Practices of Leading Organizations, *United States General Accounting Office*, GAO/AIMD-00-33, November 1999.